Undergraduate Research Opportunity Programme (UROP)

Dummy Data Generation and Data Augmentation Using Machine Learning

Omar Bin Sheik Mustafa

School of Electrical & Electronic Engineering

Academic Year 2022/23

Semester 2

Content Page

Abstract: This paper gives an overview of the challenges associated with the scarcity of data and the impacts of limited data in machine learning applications. For this study, we examine the concept of data augmentation and explore the use of Generative Adversarial Networks (GANs) and Neural Style Transfer (NST) in generating synthetic data to overcome these limitations. The paper first goes into detail about the reasons behind data scarcity and its consequences on machine learning model accuracy and performance. We then investigate various data augmentation techniques that can effectively augment current existing datasets, before exploring the capabilities of GANs and NST as efficient approaches for generating synthetic data. These techniques help to address data scarcity issues and improve the performance of the machine learning model. In the conclusion of the paper, we analyse the benefits and limitations of dummy data generation, and its potential usage in future research and work.

Keywords: Data Augmentation, Machine Learning, Generative Adversarial Networks, Neural Style Transfer

1. Introduction

Machine learning is a subfield of artificial intelligence that concentrates on the creation of algorithms and enables software applications to enhance their accuracy in forecasting results without requiring explicit programming for that purpose (Burns, 2019). In other words, machine learning involves teaching computers to recognise patterns and make predictions without being explicitly programmed for a specific task. Machine learning is grounded in statistical and mathematical principles. It utilises various techniques, such as supervised learning, which trains a model on a dataset with known inputs and outputs to enable future output predictions, and unsupervised learning, which uncovers concealed patterns or inherent structures within the input data (MathWorks, n.d.).

The emergence and rise of artificial intelligence and machine learning has revolutionised various fields of technology by enabling machines to learn from data, make inferences, and make predictions with precision. Machine learning is used every day in the real-world, with some examples being in facial recognition, social media optimisation, and predictive analytics (Tableau, n.d.). One crucial factor in the accuracy of machine learning models is the availability of large, diverse, and high-quality datasets. However, obtaining such datasets can often be time-consuming, expensive, and impractical. The major issue that comes while training machine learning models is the lack of quality and quantity of data. It is claimed by many data scientists that inadequate data, noisy data, and unclean data are exhausting the machine learning algorithms (JavaTpoint, n.d.). This issue poses a significant challenge of data scientists and engineers seeking to harness the power of machine learning.

The effectiveness and accuracy of a machine learning model is dependent on the quantity of training data that is available. The more training data the model has, the more accurate it will be. However, this implies that there exist machine learning models out there that do not have enough training data to meet their respective benchmark.

In machine learning, a large amount of data is required to train the algorithms effectively, and to ensure accurate predictions or decisions. This is because machine learning algorithms rely on patterns and relationships in data to learn. The more data available for training, the better the algorithm can learn these patterns and make more informed decisions.

2. Literature Review

2.1 Scarcity of data

Contrary to the natural assumption that nearly every business or market is inundated with a deluge of data, the reality is quite the opposite. Data is only accessible to a select group of companies, and in many instances, data is regarded as a scarce resource. In practical settings, obtaining data can prove to be rather challenging, and even if it is readily available, its quality is often uncertain (Abadi, 2021). A study done by Dimensional Research shows that about 96% of enterprises encounter data quality and labelling challenges when it comes to machine learning projects (Dimensional Research, 2019).

Acquiring large quantities of accurately labelled data can provide to be difficult for several reasons. Firstly, there is a high cost associated with data collection or annotation. Learning algorithms do not only require massive samples, but also rely on manually annotated data prior to use. This can involve intricate tasks, such as making human-like judgements on images or videos, which can demand substantial cost, time, and effort. Crowdsourcing is a method used to alleviate the cost of human labour by enlisting a wider audience to annotate data directly. However, this approach can often lead to a large amount of low-quality annotations (Zhou, 2017).

Secondly, in cases like big data analytics, privacy and security concerns prove to be an issue when it comes to obtaining data. Big data analytics is a process of examining vast quantities of complex data with the goal of uncovering concealed patterns or recognising concealed correlations. As a result of recent technological advancements, the volume of data produced by the internet, amongst other things, is experiencing a substantial increase on a day-to-day basis. To safeguard the privacy of big data, numerous mechanisms have been developed in recent years, that include data generation, storage, and processing. During the data generation phase, privacy protection measures, such as access restrictions and data falsification techniques, are employed. In the data storage phase, privacy protection strategies primarily rely on encryption techniques. Additionally, to safeguard the sensitive data, hybrid clouds are used where the data is stored in a private cloud. The data processing phase incorporates, along with extracting knowledge from the data, Privacy Preserving Data Publishing, which utilises anonymisation techniques to protect the privacy of data (Jain, 2016). In short, protection of data privacy and access limitation restricts the acquisition of data.

2.2 Impact of limited data

Overfitting is a significant issue in machine learning, especially when it comes to supervised machine learning tasks. It occurs when a machine learning algorithm adapts to the training dataset so precisely that it memorises the noise and unique characteristics of that data. As a result, the performance of the learning algorithm declines when it is tested on an unfamiliar dataset.

Conversely, underfitting, the opposite of overfitting, occurs when the machine learning model is unable to encompass the variability present in the data. This means that the model cannot capture the relationship between the input and output variables correctly or accurately, leading to high error rates on both the training set and unfamiliar data. This occurs when a model is overly simplistic and may be due to the model requiring additional training time, more input features, or reduced regularisation. Like overfitting, an underfitted model struggles to identify the prevailing trend within the data, leading to training errors and subpar model performance. If a model is unable to generalise effectively to new data, it becomes unsuitable for classification or prediction tasks. The ability of a model to generalise new data is ultimately what enables us to utilise machine learning algorithms every day to make predictions and categorise data (IBM, n.d.) (Jabbar & Khan, 2015).

2.3 Data requisites

Each machine learning project has a unique set of variables that influence the amount of training datasets necessary to achieve successful modelling. There are numerous factors which influence the size of datasets needed, such as the complexity of the model, the complexity of the learning algorithm, the labelling needs, the acceptable error margin, and input diversity. The complexity of a model refers to the quantity of parameters that the algorithm must learn, which also determines the amount of training data required. If the algorithm has more features, larger data size, and more variability of the expected output, then more input data is necessary. More complex learning algorithms require larger amounts of data. An example would be a deep learning algorithm, which is able to learn representations from raw data without the need for feature engineering. They operate without a pre-defined structure and determine all the parameters on their own. In such situations, more relevant data is necessary for the categories generated by the algorithm. As for labelling needs, it depends on how many labels the algorithm must predict. Additionally, the nature of the project being worked on is another factor that determined the quantity of data required, as each project has a distinct acceptable margin for error. Lastly, input diversity is a factor that influences the size of datasets needed for a machine learning model because in some situations, the algorithm should be able to function in unpredictable circumstances. An instance would be an AI language model that can read sentences given to it despite grammatical mistakes (Dorfman, 2022).

Having a large dataset helps to reduce biasness or randomness in the data, resulting in a more robust and reliable model. The most common way to determine if a dataset is adequate is to apply the 10 times rule, also known as the rule-of-thumb approach. This rule states that the amount of input data, that is the number of examples, should be at least 10 times greater than the number of degrees of freedom of the model. Degrees of freedom generally refers to the parameters present in the dataset. Apart from fulfilling the aforementioned rule of maintaining a suitable ratio between the amount of input data and the number of degrees of freedom, it is crucial to have appropriate coverage across various categories or classes within a dataset to avoid class imbalance or sampling bias issues (Dorfman, 2022) (Smolic, 2022).

2.4 Dummy data generation and data augmentation

2.4.1 Dummy data

Dummy data, also known as synthetic data, refers to artificially generated mock data used as a replacement for live data in testing scenarios. Dummy data serves as a stand-in for live data, which is only introduced once it is confirmed that the trial program does not adversely affect the underlying data in any unintended manner (Noah, n.d.). Essentially, dummy data is created to fill gaps in datasets and enhance the performance of machine learning models when actual data is unavailable or limited. It should be noted that the dummy data has similar characteristics to the live data that it is supposed to mimic, so that the accuracy and performance of the machine learning model remain relevant.

2.4.2 Overview of data augmentation

To build relevant and useful machine learning models, the validation error should decrease according to the training error. To achieve this, data augmentation can be utilised. Through data augmentation, the expanded dataset will encompass a wider range of potential data points, therefore reducing the gap between the training and validation sets, as well as any forthcoming test sets. Data augmentation aims to combat overfitting issues by addressing the source of the problem, which is the training dataset. It operates on the assumption that additional information can be extracted from the original dataset through augmentations. Through methods like data warping, these augmentations artificially increase the size of the dataset. Data warping augmentations modify existing images while keeping their labels preserved (Shorten & Khoshgoftaar, 2019).

In this section, we will explore a manual form of data augmentation via the method of data warping. Data warping encompasses a variety of transformation methods that modify existing images in our dataset to produce new and relevant augmented data. We will explore several ways to apply warping in images, including rotation, warp shift, and the addition of Gaussian noise. The objective of this experiment is to analyse how data augmentation, via warping, can increase the diversity of the dataset and subsequently create a more generalised machine learning model.

We will be using the MNIST database to illustrate an example of data augmentation that is used in image processing. The Modified National Institute of Standards and Technology (MNIST) database is a vast collection of handwritten digits, comprising of a training set of 60,000 examples and a test set of 10,000 examples. It is a subset of the NIST Special Database 3 (written by employees of the United States Census Bureau) and 1 (written by high school students), which contain monochrome images of handwritten digits the images of the digits were normalised to a fixed size and centred. The original black and white images from NIST were first size-normalised to fit in a 20x20 pixel box while maintaining their aspect ratio, resulting in grey levels due to the anti-aliasing technique used in the normalisation algorithm. The images were then centred in a 28x28 image by determining the centre of mass of the pixels and shifting the image so that this point is at the centre of the 28x28 field (LeCun et al., 1998). Figure 1 shows some examples of handwritten digits that are available in the MNIST database.

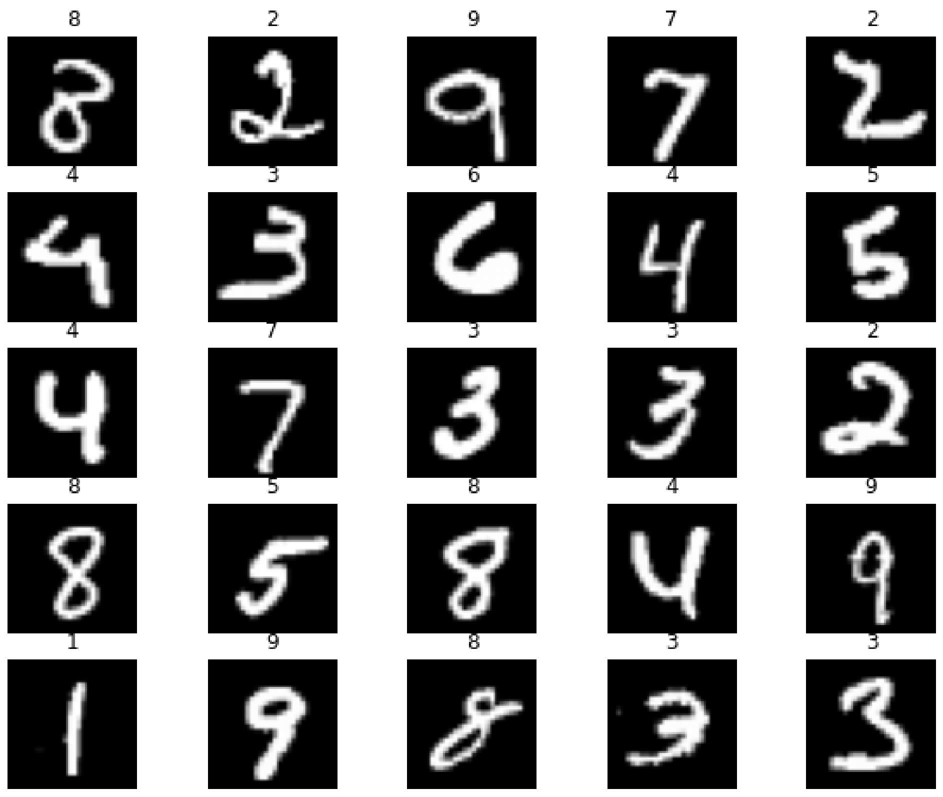


Figure 1: Some handwritten digits in the MNIST database

For the purposes of this demonstration, we will be using the first digit that is in the dataset, which is of a number '5', as shown in Figure 2.

```
<matplotlib.image.AxesImage at 0x1bc99e8c070>
```

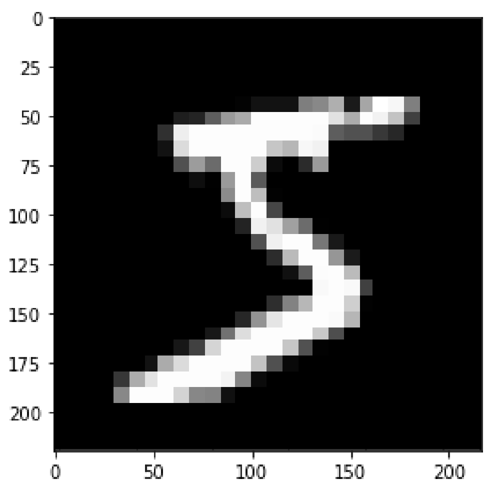


Figure 2: First digit in the MNIST dataset, which depicts a handwritten '5'.

2.4.3 Rotation of the image

Image rotation is a commonly utilised augmentation technique in image processing. Rotation augmentations involve rotating an image to the right or left about an axis between 1^o and 359^o. The safety and effectiveness of rotation augmentations depend heavily on the degree of rotation parameter used. Small rotations such as between -20^o to 20^o may prove useful, especially in digit recognition tasks like this MNIST dataset. However, it should be noted that as the degree of rotation increases, the label of the data is no longer preserved after transformation, thus reducing the usefulness of the augmentation (Shorten & Khoshgoftaar, 2019). As shown in Figure 3, we rotate the image of the '5' by 45^o anti-clockwise.

```
Rotated Image
<matplotlib.image.AxesImage at 0x1bc99ecc1c0>
```

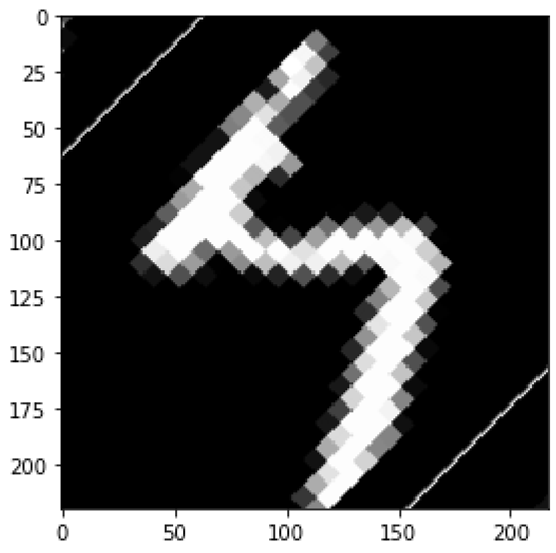


Figure 3: A rotated version of the image

2.4.4 Wrap Shift

It is possible to encounter scenarios where the object in the image is not centralised. In such cases, shifting the image can add shift-invariance to the image. By translating the original image in a specific direction, the empty space can be filled with a constant value such as 0 or 255 or filled with random or Gaussian noise. This padding preserves the image's spatial dimensions after augmentation. This will lead to a more generalised model (Shorten & Khoshgoftaar, 2019) (Sharma, 2019). In the following example, shown in Figure 4, we shift the image by 25 pixels in both the x-direction and y-direction.

```
Text(0.5, 1.0, 'Wrap Shift')
```

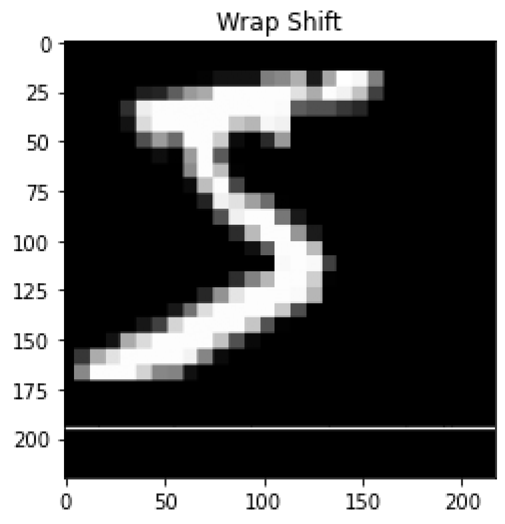


Figure 4: A shifted version of the image

2.4.5 Gaussian Noise injection

Noise injection involves adding a matrix of random values typically drawn from a Gaussian distribution. Incorporating noise into images can aid convolutional neural networks (CNNs) to acquire more robust features (Shorten & Khoshgoftaar, 2019). For this experiment, we take the standard deviation of the noise added to be 0.155. Increasing this value will add more noise to the image, while decreasing this value adds less noise to the image. Augmenting more digits in the dataset will result in a more robust model.

```
Text(0.5, 1.0, 'Random Noise')
```

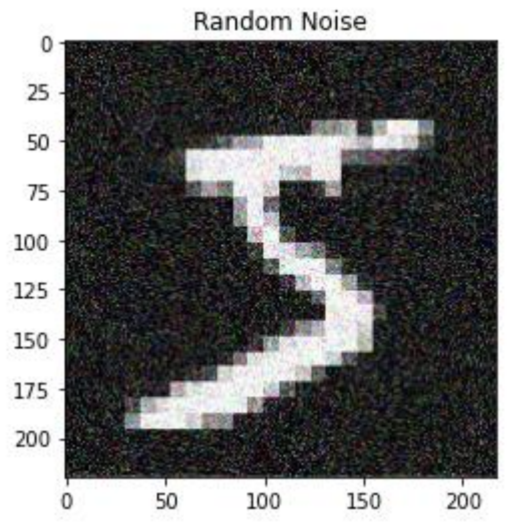


Figure 5: The image after adding Gaussian, or Random, noise.

3. Data augmentation using machine learning

3.1 Generative Adversarial Networks (GANs)

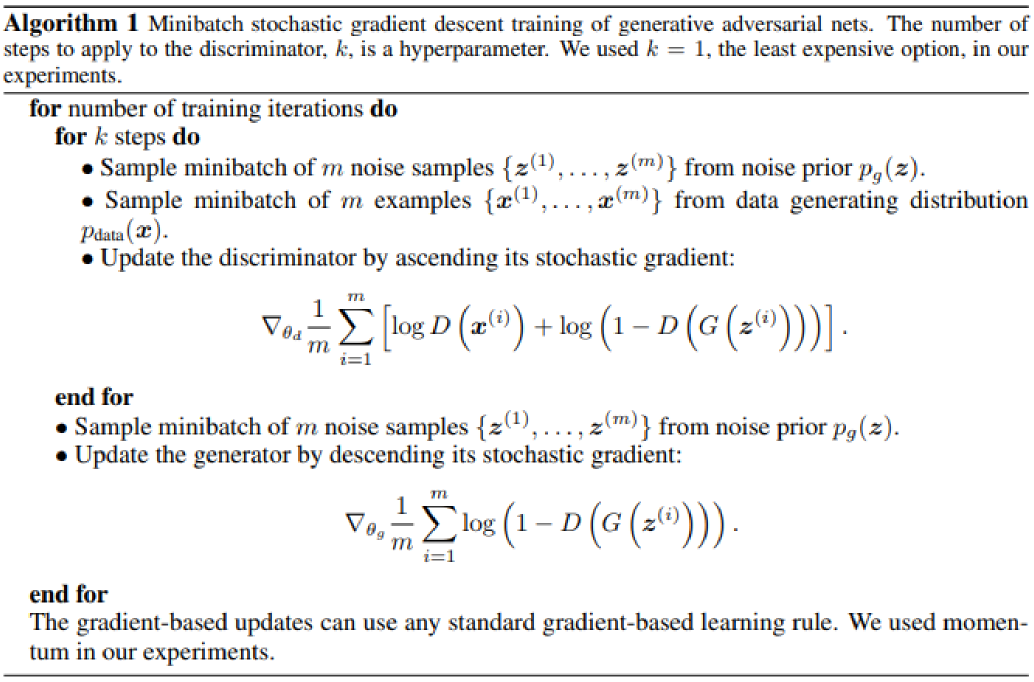
3.1.1 Overview

GANs represent a technique for generative modelling using deep learning methodologies, like convolutional neural networks, to create new data samples (Brownlee, 2019). They were introduced in 2014 by Ian Goodfellow and his collaborators (Giles, 2018).

GANs comprise of two models: a generator and a discriminator. Although the two models are typically implemented by neural networks, they can also be implemented by using any differentiable system that maps data from one space to another. The objective of the generator is to capture the distribution of genuine examples for new data example generation. The discriminator, on the other hand, is typically a binary classifier, tasked with distinguishing between generated and real examples as accurately as it can. The optimisation of GANs represents a minimax optimisation problem. The optimisation stops at a saddle point that serves as a minimum with respect to the generator, and a maximum with respect to the discriminator. In other words, the optimisation aims to achieve Nash equilibrium. At this point, the generator can be considered to have successfully captures the authentic distribution of real examples (Gui et al., 2020). For the context where the generator and the discriminator are represented by multilayer perceptions, the fundamental structure is always established with backpropagation. The technique used is to get the maximum probability of assigning accurate labels to both training models and samples from the generator, while also concurrently training it (Singh et al., 2021). The equation shown below is the combined loss function, and the following algorithm in Figure 7 is used to train GANs, both derived by Goodfellow et al. in their original paper.

$$\min_{G} \max_{D} V(D,G) = \mathbb{E}_{\chi \sim p\mathrm{data}(x)}[\log \mathrm{D}(x)] + \mathbb{E}_{\mathrm{z} \sim p\mathrm{z}(z)}[\log(1 - \mathrm{D}(\mathrm{G}(\mathrm{z})))]$$

Figure 6: Combined loss function



Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

```
for number of training iterations do
    for $k$ steps do
        • Sample minibatch of $m$ noise samples $\{z^{(1)}, \ldots, z^{(m)}\}$ from noise prior $p_g(z)$.
        • Sample minibatch of $m$ examples $\{x^{(1)}, \ldots, x^{(m)}\}$ from data generating distribution $p_{\text{data}}(x)$.
        • Update the discriminator by ascending its stochastic gradient:
              $\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[ \log D\left(x^{(i)}\right) + \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right) \right].$
    end for
    • Sample minibatch of $m$ noise samples $\{z^{(1)}, \ldots, z^{(m)}\}$ from noise prior $p_g(z)$.
    • Update the generator by descending its stochastic gradient:
          $\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right).$
end for
The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.
```

Figure 7: Algorithm used to train GANs using minibatch stochastic gradient descent (Goodfellow et al., 2014)

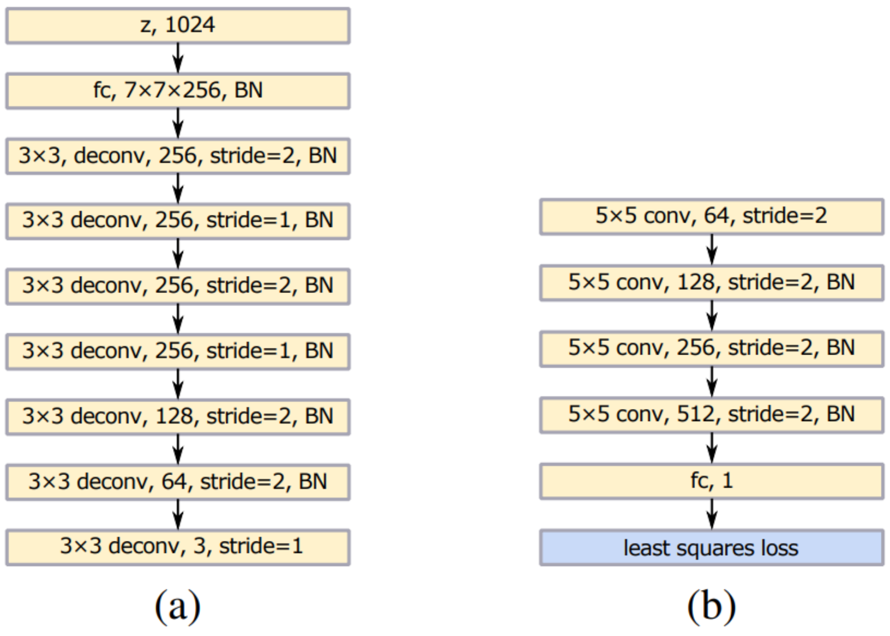

z, 1024
fc, 7×7×256, BN
3×3, deconv, 256, stride=2, BN
3×3 deconv, 256, stride=1, BN
3×3 deconv, 256, stride=2, BN
3×3 deconv, 256, stride=1, BN
3×3 deconv, 128, stride=2, BN
3×3 deconv, 64, stride=2, BN
3×3 deconv, 3, stride=1
(a)
5×5 conv, 64, stride=2
5×5 conv, 128, stride=2, BN
5×5 conv, 256, stride=2, BN
5×5 conv, 512, stride=2, BN
fc, 1
least squares loss
(b)

Figure 8: Model architecture of a GAN. "k x k conv/deconv, C, stride=S" represents a convolutional/deconvolutional layer with a (k x k) kernel, C output filters, and a stride of S. BN signifies that a batch normalisation layer follows the layer with BN. "fc, N" refers to a fully connected layer with N output nodes. The activation layers have been excluded for clarity. The left (a) is the generator, while the right (b) is the discriminator (Mao et al., 2017).

3.1.2 Illustrating GAN using MNIST dataset

In this section, we present an illustration of a GAN using the MNIST dataset and programming the GAN using the PyTorch framework. After preparing the dataset, the training images were plotted and subsequently labelled "Training Images", as shown in Figure 9. The images chosen were completely random and some are shown in the figure below.

Figure 9: Training images used to create the GAN.

After that, weight initialisation was conducted for the neural network. Weight initialisation is a process of assigning small random values to the weights of a neural network which serve as the starting point for the optimisation, or the learning and training, of the neural network model (Brownlee, 2021). Then, the generator was composed, and subsequently the discriminator. The generator model consists of 5 layers, that are the input layer, 3 hidden layers, and the output layer. The input layer and hidden layers consist of a strided two-dimensional convolutional transpose layer, a two-dimensional norm layer, and a ReLU activation. The output layer, on the other hand, consists of a strided two-dimensional convolutional transpose layer and a tanh function. A strided convolutional transpose layer helps control the spacing between the output values when up sampling the input data. The discriminator model consists of 5 layers as well; the input layer, 3 hidden layers, and the output layer. The input layer consists of Conv2d and LeakyReLU. The hidden layers are just like the input layer, except they also consist of BatchNorm2d. The output layer, lastly, consists of Conv2d and a sigmoid function.

Hence, the generator and discriminator were created. A loss function and optimiser were set up, with a binary cross entropy loss function (torch.nn.BCELoss()) used for the loss function and ADAM (torch.optim.Adam()) used for the optimiser. The GAN model was then trained.

The process of training the discriminator is done by giving the network labelled images coming from the generator, which are fake, and from the training data, which are real. The discriminator learns to classify between real and fake images with a sigmoid prediction output. The objective of this training is to get the highest probability of correctly classifying a given input as real or fake.

The process of training the generator is done by requiring the discriminator to inform it whether it is generating fake images well. Thus, the combined network is created to train the generator, consisting of both generator and discriminator models. The objective of training the generator is to achieve the generation of better fake images.

The real images and fake images were plotted, as shown in the figures below. The fake images were constructed by the GAN, after training it for 5 epochs.

Figure 10: Real Images used in training.

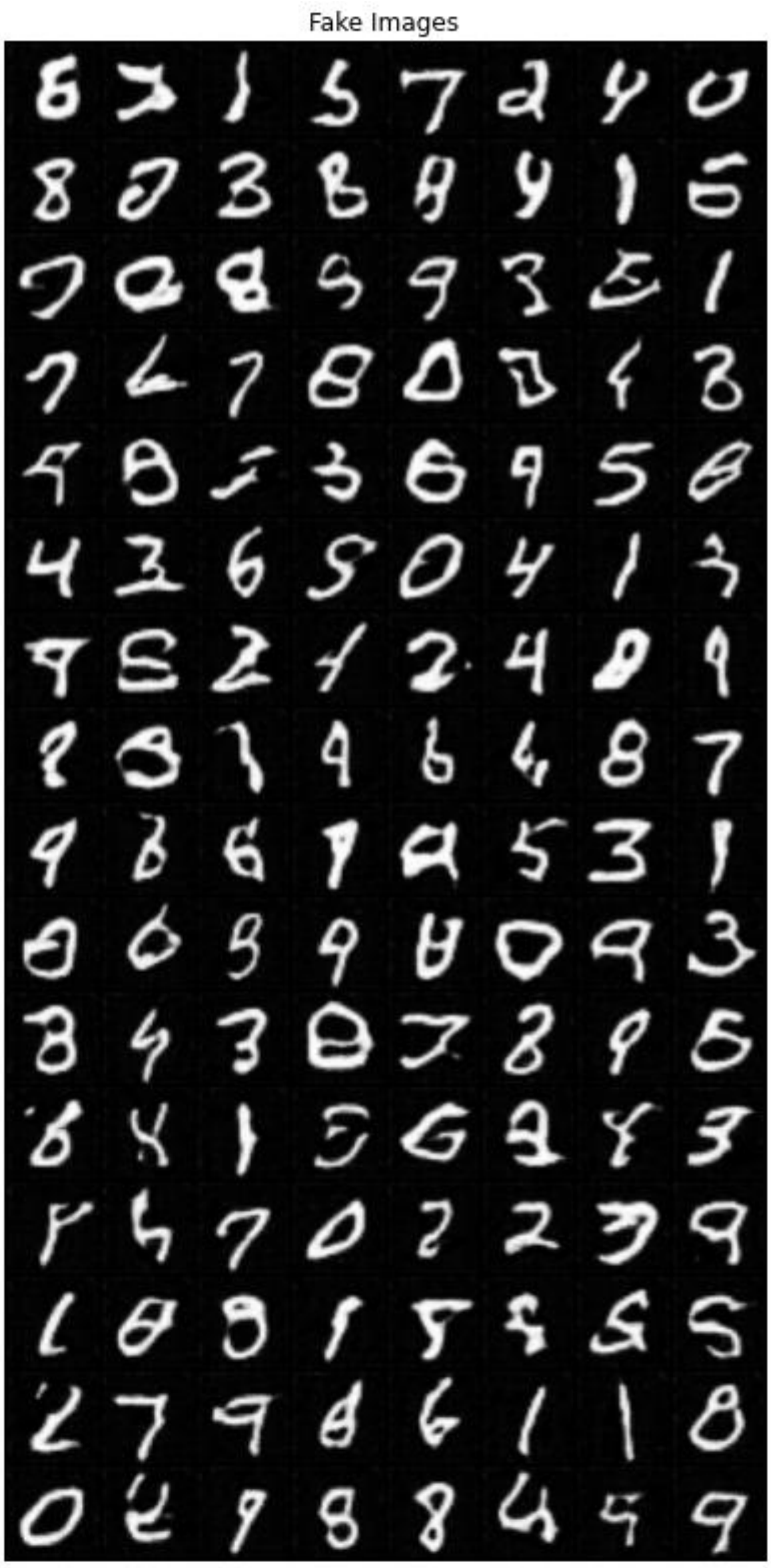


Figure 11: Fake images generated by the GAN.

3.2 Neural Style Transfer (NST)

3.2.1 Overview

NST is a technique used in deep learning and computer vision that combines the content of one image with the style of another image. NST represents a category of software algorithms that utilise neural networks to transform scenes or modify the environment of a media. NST is widely used in image and video editing software, allowing image stylisation based on a general model, distinguishing it from traditional methods. As a result, NST has gained popularity in the entertainment industry, as professional editors and media producers create media quicker and provide the public with recreational use (Singh et al, 2021).

By utilising NST, we can create a new image that preserves the content, or structure, of the original image while integrating the style of the second image into it.

3.2.2 Implementation of NST to augment data

In this section, we explore using NST for the use of data augmentation, particularly in Computer Vision. We will utilise two images, a style image and a content image, and mix them together to produce an output image.

Shown below in Figure 12 are the style image and content image.

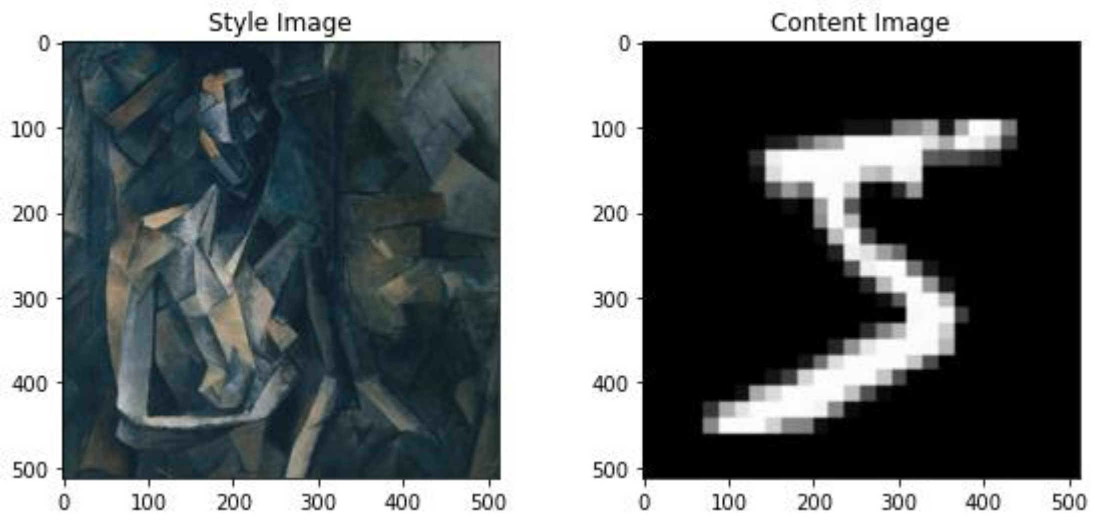


Figure 12: Style image (left) and Content image (right)

First, the loss functions are created. There are two loss functions: content loss and style loss. The content loss is calculated as the distance between the features of the base image and the features of the combination image, while ensuring that the generated image is sufficiently similar to the original one (Chollet, 2020). Style loss is implemented similarly to the content loss module. It will function as a transparent layer in a network, calculating the style loss of that specific layer. For the purpose of this illustration, we import a pre-trained neural network from PyTorch's models. PyTorch's implementation of VGG is a module that consists of two Sequential modules: 'features', which contains convolution and pooling layers, and 'classifier', which contains fully connected layers. We will only use the 'features' module since we require the output of individual convolution layers to compute content and style loss. We use the L-BFGS algorithm from the PyTorch library to run our gradient descent. Unlike training a network, we would prefer to train the input image to minimise the content and style losses. Lastly, we defined a function that performs the neural transfer. During each iteration of the network, a revised input is provided, and the network computes new losses (Jacq, n.d.). Then, finally, the algorithm was run. Shown in Figure 13 is the resultant output image.

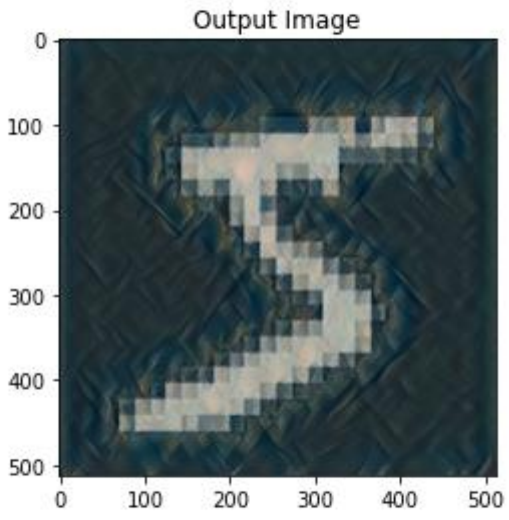


Figure 13: Output image of NST

4. Discussion

4.1 Overview

Synthetic data generation using GANs and NST proves to be useful in minimising data scarcity. The ability of GANs to learn nearly any data distribution patterns from a given training data makes them extremely potent and adaptable to changes. They have given rise to several new ground-breaking deep learning applications that were previously unachievable. Applications include medical imaging, game development, image enhancements, cybersecurity, and fashion and advertising (Sambhav, n.d.). GANs show significant potential in creating various forms of media. They were one of the most widely used generative AI techniques until the emergence of the Transformer by a team of Google researchers in 2017.

Transformer models are a type of deep learning architecture that gains an understanding of context and significance by observing the connections between elements in sequential data. The Transformer is the first sequence transduction model that relies entirely on attention, replacing the traditional recurrent layers used in encoder-decoder architectures with multi-headed self-attention. In the case of translation tasks, the Transformer can be trained at a much faster pace as compared to architectures that rely on recurrent or convolutional layers (Vaswani et al., 2017). Transformers are a fundamental technology that instilled various breakthroughs in large language models, such as Generative Pre-trained Transformers (GPTs). They are currently being applied to multimodal AI tasks capable of efficiently correlating a wide range of content, including text, image, audio, and robot instructions across various media types, surpassing even the GANs (Lawton, 2023).

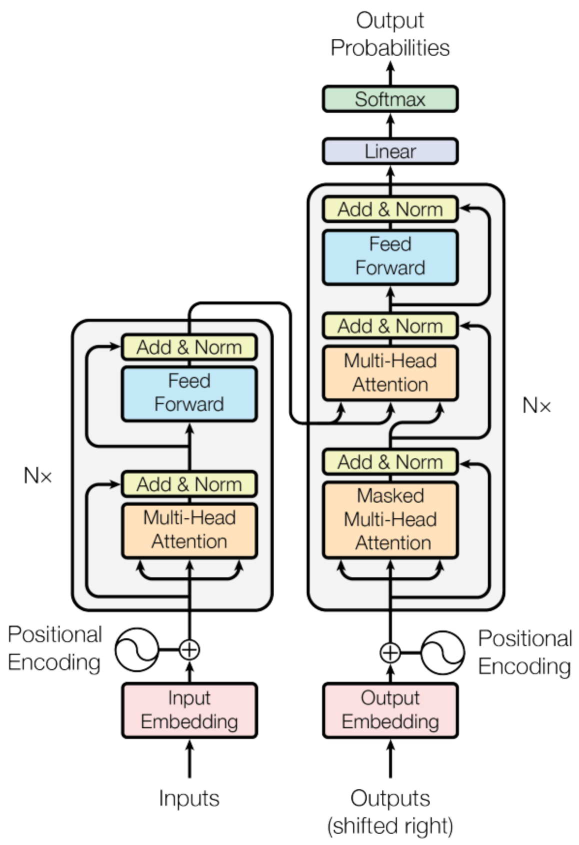


Figure 14: Model architecture of the Transformer.

Shown above in Figure 14 is a diagram of the model architecture of the Transformer. Many competitive neural sequence transduction models employ an encoder-decoder architecture [5, 2, 35]. In this structure, the encoder converts an input sequence of symbol representations (x1, … xn) into a continuous representation sequence z= (z1, … zn). Given z, the decoder generates an output sequence (y1, … ym) one symbol at a time. The model is auto regressive at every step, using previously generated symbols as an extra input to produce the next symbol. The Transformer model adheres to this overall architecture, utilising stacked self-attention and point-wise, fully connected layers for both the encoder and decoder (Vaswani et al., 2017).

4.2 Benefits of using machine learning to augment data

Using machine learning to produce dummy data as a data augmentation technique can provide several advantages over older and more manual methods of acquiring data. For example, GANs can operate with little supervision. After the initial input, the GAN continues to train itself by coming up with its own training data. Also, due to a GAN's capacity for independently replicating data distribution, it can efficiently generate highly specialised collections of data. This hence reduces the amount of manual or human labour needed (Lombardi, 2022).

4.3 Limitations of using machine learning to augment data

There are some challenges that arise when using machine learning to augment data. For instance, creating a GAN model is not so simple. Developing one requires extensive technical expertise and sophisticated datasets. Therefore, to construct a precise model, an experienced developer is advised (Lombardi, 2022). Meanwhile, as for using NST in data augmentation, there is a risk of losing vital content information. When applying style transfer to an image, crucial information that we need from the image could be lost in the process. This will affect the performance of the model.

5. Conclusion

Performing dummy data generation and data augmentation using machine learning can be an efficient way of creating more data in cases where it is challenging to acquire data. Using GANs, we can easily augment data via the synthetic data that our GANs produce. With NSTs, we can also easily double our quantity of data with just one style image applied.

However, the invention of the transfer was a game-changer in the field of AI development. Transformers, since their emergence, has since led to significant advancements across numerous domains, including natural language processing, computer vision, and reinforcement learning. For instance, ChatGPT, a widely used AI language model now, is based on the GPT architecture, a transformer model. It has even shown the capability to assist in text data augmentation (Dai et al., 2023).

It is probable that in the future, the large transformer models like GPT will also be commonly used for data augmentation techniques. However, there remains the concern of whether or not the data generated by these models are actually helping to improve the training performances of machine learning models with less error. A possible future for GANs and transformers would be that combining the strengths of GANs and transformers can lead to more relevant and diverse data generation and augmentation, which will hence ensure that the machine learning model's performance is not jeopardised.

This paper has provided an in-depth investigation on the reasons and impacts of the scarcity of data, and the different techniques used to generate dummy data and augment data using machine learning. We have delved into the realm of deep learning where we explored the use of GANs and NST to generate synthetic data. Moreover, we discussed the potential of these models, together with transformers, in becoming more efficient data augmentation techniques.

6. References

Abadi, A. (2021). *A survey on data-efficient algorithms in big data era.* https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00419-9

Brownlee, J. (2019). *A Gentle Introduction to Generative Adversarial Networks (GANs).* https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/

Brownlee, J. (2021). *Weight initialization for deep learning neural networks.* https://machinelearningmastery.com/weight-initialization-for-deep-learning-neural-networks/#:~:text=Weight%20initialization%20is%20a%20procedure,of%20the%20neural%20network%20model

Burns, E. (2019). *What is machine learning and why is it important?* https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML

Chollet, F. (2020). *Neural style transfer.* https://keras.io/examples/generative/neural_style_transfer/#:~:text=The%20content%20loss%20is%20a,enough%20to%20the%20original%20one

Dai, H., Liu, Z., Liao, W., Huang, X., Cao, Y., Wu, Z., Zhao, L., Xu, S., Liu, W., Liu, N., Li S., Zhu, D., Cai, H., Sun, L., Li, Q., Shen, D., Liu, T., Li, X. (2023). *AugGPT: Leveraging ChatGPT for text data augmentation.* https://arxiv.org/abs/2302.13007

Dimensional Research (2019). *Artificial Intelligence and Machine Learning Projects are obstructed by Data Issues.* https://cdn2.hubspot.net/hubfs/3971219/Survey%20Assets%201905/Dimensional%20Research%20Machine%20Learning%20PPT%20Report%20FINAL.pdf

Dorfman, E. (2022). *How much data is required for machine learning?* https://postindustria.com/how-much-data-is-required-for-machine-learning/

Giles, M. (2018). *The GANfather: The man who's given machines the gift of imagination.* https://www.technologyreview.com/2018/02/21/145289/the-ganfather-the-man-whos-given-machines-the-gift-of-imagination/

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y. (2014). *Generative Adversarial Nets.* https://arxiv.org/abs/1406.2661

Gui, J., Sun, Z., Wen, Y., Tao, D., Ye, J. (2020) *A Review on Generative Adversarial Networks: Algorithms, Theory, and Applications.* https://arxiv.org/abs/2001.06937

IBM (n.d.). *What is underfitting?*

https://www.ibm.com/topics/underfitting#:~:text=the%20next%20step-,What%20is%20underfitting%3F,training%20set%20and%20unseen%20data.

Jabbar, H. K., Khan, R. Z. (2015). *Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study).* https://d1wqtxts1xzle7.cloudfront.net/37902828/017-libre.pdf?1434274012=&response-content-disposition=inline%3B+filename%3DMETHODS_TO_AVOID_OVER_FITTING_AND_UNDER.pdf&Expires=1682848924&Signature=FcNqQZKtgXLGUI6DJjLFf-u~QPS1xIgDAMZCEyMHg3RySXS~uxXl3hI-216xlFI26b~fIjt5Ddv54G9Sc-UA-O2QP61dMGBB76Aqr3dflH3QFAKfb12dubn8OwATjJ0sx5P2LD1coqPqsJVPRB8l~LKEz6vkcV3ElsyUPMIYh2Q9udeCEubWqMIfNdu00xQIEbTUPSbPzfB86PcBc84x~Wabr65gzkxkE19YWYDGq8um3vvy~WSBaWT0KeIMUk0rLWJ5~MgQ0RFPrZzSG01357DGhEFgupD1aUv-SSHdkgvURGZwlWhe5sdPS1TCHXWEVIJd2Hw8f0ObgH2vkw4M0g__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA

Jain, P., Gyanchandani, M., Khare, N. (2016). *Big data privacy: a technological perspective and view.* https://journalofbigdata.springeropen.com/articles/10.1186/s40537-016-0059-y#Sec2

JavaTpoint (n.d.). *Issues in Machine Learning.* https://www.javatpoint.com/issues-in-machine-learning

Lawton, G. (2023). *GAN vs. transformer models: Comparing architectures and uses.* https://www.techtarget.com/searchenterpriseai/tip/GAN-vs-transformer-models-Comparing-architectures-and-uses

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P. (1998). *Gradient-based learning applied to document recognition.* https://paperswithcode.com/paper/gradient-based-learning-applied-to-document

Lombardi, P. (2022). *What is GAN Machine Learning and what are its benefits?* https://www.indeed.com/career-advice/career-development/what-is-gan-machine-learning

Mao, X., Li, Q., Xie, H., Lau, R. Y. K, Wang, Z., Smolley, S.P. (2017). *Least Squares Generative Adversarial Networks.* https://openaccess.thecvf.com/content_iccv_2017/html/Mao_Least_Squares_Generative_ICCV_2017_paper.html

MathWorks (n.d.). *What is Machine Learning?* https://www.mathworks.com/discovery/machine-learning.html#:~:text=Machine%20learning%20uses%20two%20types,intrinsic%20structures%20in%20input%20data

Noah (n.d.). *Dummy Data: Definition, Example & How to Generate It.*
https://analystanswers.com/dummy-data-definition-example-how-to-generate-it/

Sambhav, K. (n.d.). *GANs and their applications.*
https://nusit.nus.edu.sg/technus/gans-and-their-applications/

Sharma, P. (2019). *Image augmentation for deep learning using PyTorch – Feature engineering for images.*
https://paperswithcode.com/paper/gradient-based-learning-applied-to-document

Singh, A., Jaiswal, V., Joshi, G., Sanjeeve, A., Gite, S., Kotecha, K. (2021). *Neural Style Transfer: A Critical Review.*
https://ieeexplore.ieee.org/document/9539183

Smolic, H. (2022). *How much data is needed for machine learning?*
https://graphite-note.com/how-much-data-is-needed-for-machine-learning

Tableau (n.d.). *Real-world examples of machine learning (ML).*
https://www.tableau.com/learn/articles/machine-learning-examples

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., Polosukhin, I. (2017). *Attention is all you need.*
https://arxiv.org/pdf/1706.03762.pdf

Zhou, Z. (2018). *A brief introduction to weakly supervised learning.*
https://academic.oup.com/nsr/article/5/1/44/4093912?login=false